Ссылка для цитирования этой статьи:

Бевзюк А.А., Бевзюк И.К., Цветкова С.Е. Сравнительный анализ способов оценки работы командами разработки // Human Progress. 2023. Том 9, Вып. 2. С. 18. URL: http://progress-human.com/images/2023/Tom9_2/Bevzuk.pdf. DOI 10.34709/IM.192.18. EDN ZIOMBQ.

УДК 007.3

СРАВНИТЕЛЬНЫЙ АНАЛИЗ СПОСОБОВ ОЦЕНКИ РАБОТЫ КОМАНДАМИ РАЗРАБОТКИ

Бевзюк Антон Анатольевич
Engineering Manager
ООО «Майндбокс»

bevzuk@gmail.com
26, ул Правды
г. Москва, Россия, 125124
+7 (495) 921-36-44

Бевзюк Ирина Константиновна
старший преподаватель кафедры Теории и практики иностранных языков и лингводидактики
ФГБОУ ВО «Нижегородский государственный педагогический университет им. Козьмы Минина»

ibevzuk@gmail.com
1, ул. Ульянова
г. Нижний Новгород, Россия, 603950
+7 (831) 262-20-44

Цветкова Светлана Евгеньевна
кандидат педагогических наук, доцент кафедры Теории и практики иностранных языков и лингводидактики
ФГБОУ ВО «Нижегородский государственный педагогический университет им. Козьмы Минина»

svetlanatsvetkova5@gmail.com
1, ул. Ульянова
г. Нижний Новгород, Россия, 603950
+7 (831) 262-20-44

Аннотация. Каждой команде разработчиков важно правильно оценивать свою работу, чтобы достигать поставленных целей и укладываться в сроки. Существует несколько методов оценки работы, каждый из которых имеет свои преимущества и недостатки. В данной статье авторы анализировали команды, которые используют абсолютные оценки в часах, чтобы более точно определить количество времени, необходимое на выполнение тех или иных задач. Было выявлено, что это может требовать большого количества времени при планировании. Схожие

недостатки были определены для метода абсолютной оценки в идеальных часах. Далее были проанализированы относительные оценки, в частности, относительная оценка в сторипоинтах позволяет более быстро оценить сложность задач и использовать этот показатель в качестве единицы измерения, но не всегда удобна для больших проектов и требует определенного навыка оценивания работы в относительных единицах измерения. Также был проанализирован безоценочный подход #noestimates, который подразумевает отказ от оценок вообще, что может быть полезным для команд, которые считают оценки неэффективными или которые не могут оценить свою работу в часах или сторипоинтах. Исследование может быть интересно руководителям команд, которые выбирают метод оценки в зависимости от особенностей проекта и опыта команды.

Ключевые слова: Agile; менеджмент; планирование; способы оценки; абсолютная оценка; относительная оценка; сторипоинты; #noestimates.

JEL коды: M12; O21.

Введение

Очевидно, что в век всеобщей компьютеризации и информатизации общества труд программиста высоко востребован. Качество и продуктивность рынка услуг программирования во многом определяет успешное функционирование других отраслей производства и социальных сфер общества. Существует множество способов, при помощи которых команды разработки оценивают свою производительность и планируют работу**.**

Целью этой статьи является исследование преимуществ и недостатков разных способов оценки: абсолютной оценки в часах, абсолютной оценки в идеальных часах, относительной оценки в сторипоинтах и #noestimates.

1. Материалы и методы

Теоретическую базу исследования составили труды зарубежных исследователей (Mike Cohn [1; 2], Vasko Duarte [3], Mary Igbal [4; 5], Ron Jeffries [6], Jeff Sutherland [7] и пр.).

Эмпирические методы основаны на применении и апробации разных способов оценки производительности и планирования труда в реальных условиях управления работой команды разработчиков.

О точности оценки

Прежде чем говорить о способах оценки, стоит обратить внимание на частое неверное понимание смысла оценки. Когда разработчики дают оценку, это всегда *прогноз* того, что случится в будущем. Как правило, разработка ведется в условиях высокой неопределенности

как со стороны бизнеса, так и со стороны разработки: могут измениться ожидания заказчика, или в процессе работы разработчику может прийти другая идея реализации. Поэтому невозможно предсказать сроки выполнения задачи с высокой точностью, можно говорить лишь о вероятностных оценках. Эту проблему подробно исследовал в своих работах Майк Кон. Оценка – это всегда случайная величина со несимметричным распределением (оценка может быть сколько угодно большой, но всегда больше 0), а значит она всегда неточна по определению. Требовать от любой оценки высокой точности невозможно, так как любая оценка – это прогноз, а видеть будущее мы пока еще не научились [1; 2]. Точная оценка – это оксюморон.

Тем не менее, раз за разом менеджеры команды разработки совершают ошибку, принимая оценку команды за обещание выполнить работу в указанные сроки. На эту проблему указывал Ron Jeffries [6; 7], которому приписывают изобретение сторипоинтов. И если команда не укладывается в прогнозное значение, к ней начинают применять методы административного давления. Такое поведение менеджеров неизбежно влияет на оценку: команды будут стремиться к завышению оценки, чтобы избежать неприятных последствий. Если же оценку воспринимать как статистическую случайную величину, с ней становится проще работать: ведь мы прекрасно понимаем, что точно предсказать значение невозможно. Командам и менеджерам необходимо выработать для себя удобный и практичный способ прогнозирования работы, чтобы совместно приходить к результату, а не заниматься оценкой ради оценки [8].

Имеющийся практический опыт и теоретический обзор источников по проблеме показали, что проблема оценки производительности и планирования работы не до конца исследована. Итак, перейдем к описанию разных способов оценки и их особенностей.

2. Результаты исследования и их обсуждение

Абсолютная оценка в часах.

При этом способе оценки команда дает прогноз, сколько времени (в часах, днях, неделях, месяцах) займет та или иная работа. Такой способ оценки встречается довольно часто и это легко объяснить. Все мы с детства привыкли оценивать длительность работы в часах: “Сколько времени тебе нужно, чтобы сделать домашнее задание по математике?”. Преимущество абсолютных оценок в часах в том, что с них легко начать, ведь с ними все знакомы с детства. Но за этой легкостью кроются неочевидные сложности.

Во-первых, такая оценка всегда субъективна и сильно зависит от экспертизы разработчика. Одну и ту же задачу опытный разработчик может сделать за 2 часа, в то время

как начинающему программисту понадобится 3-4 дня. Даже среди опытных разработчиков разброс во времени выполнения одной и той же задачи может отличаться в разы или даже десятки раз в зависимости от особенностей задачи, экспертизы в конкретной области и индивидуальных особенностей.

Во-вторых, оценка в часах усложняет планирование. Как планировать работу, если все задачи оценил опытный разработчик, а команда состоит из 2 опытных и 5 неопытных? Приходится или вводить какие-то коэффициенты, отражающие разницу в опыте, или планировать работу по людям, то есть подробно распределять кто, когда и какую задачу будет делать, а затем балансировать нагрузку между участниками команды. Такой способ планирования очень ненадежен, ведь при любом сдвиге сроков весь план начинает сыпаться.

В-третьих, ни один разработчик не работает непрерывно 8 часов в день. Если разработчик даст оценку 8 часов, в нее будет входить не только непосредственно разработка, но и перерывы на поиск оптимального решения, обсуждение с коллегами, написание кода и тестов, дебаг и исправление ошибок, получение обратной связи, перерывы на обед и отдых, а также митинги, встречи, тренинги и прочие отвлечения. Индивидуальные физиологические особенности влияют на производительность отдельного человека в течение дня. Поэтому абсолютная оценка в часах довольно «шумная».

Абсолютная оценка в идеальных часах.

В попытке справиться с третьим недостатком, была придумана оценка в идеальных часах. Давая такую оценку, программист предполагает, что он будет выполнять задачу в “идеальных” условиях, в которых программист:

- располагает всей необходимой информацией и знаниями;
- с первого раза пишет оптимальный код без ошибок;
- всегда работает с одинаково высокой производительностью, не отдыхает и не болеет.

Такая оценка позволяет избавиться от шума и сделать оценку более сравнимой, но она наследует основные недостатки всех абсолютных оценок: несравнимость оценок, данных разработчиками с разными компетенциями и, как следствие, усложнение процесса планирования.

Относительная оценка в сторипоинтах.

Оценивая работу в сторипоинтах, команда использует относительную шкалу оценок и сравнивает сложность задач между собой, не прибегая к оценкам в абсолютных величинах (часах, днях, неделях). Часто для оценивания используется последовательность Фиббоначи (1, 2, 3, 5, 8, 13, 21) или похожая (1, 2, 3, 5, 8, 13, 20, 50, 100). Давайте разберемся как это работает на примере.

Допустим, у нас есть команда, которая умеет копать ямы. В команде есть неопытный рабочий с детским совочком и опытный рабочий с экскаватором. Недавно команда выкопала яму размером 1x1x1 метр. Предположим, что команда оценила эту работу в 1 сторипоинт (абстрактная единица сложности работы). Команде нужно выкопать траншею размером 1x1x3 метра. Очевидно, эта работа в 3 раза сложнее предыдущей. Поэтому команда оценивает ее в 3 сторипоинта.

Обратите внимание, что даже при наличии разницы в компетенциях, относительная оценка будет одинаковой и для опытного, и для неопытного рабочего. Если бы мы оценивали в абсолютных величинах (в часах), то если неопытный рабочий с детским совочком на первую яму потратил 8 часов, то на вторую он потратит 24 часа. Аналогично если опытный рабочий на экскаваторе выкопал первую яму за 5 минут, то на вторую у него уйдет приблизительно 15 минут. Но у обоих рабочих вторая яма в 3 раза сложнее первой. Таким образом, даже при разных абсолютных оценках мы получаем одинаковые относительные оценки.

В относительную оценку в сторипоинтах команда включает три составляющих: объем, сложность и риски [4; 5]. Объем мы уже рассмотрели на примере двух ям, одна из которых больше другой в 3 раза. При одинаковом объеме работы может отличаться сложность. Например, выкопать яму одинакового размера в каменистом грунте сложнее, чем в мягкой почве. Задача с большей сложностью будет иметь большую относительную оценку. Если мы копаем яму в месте, где проложен электрокабель, то есть риск его порвать, и тогда нам придется тратить дополнительные усилия на его восстановление. Подобные риски также увеличивают оценку.

Как же использовать относительные оценки для прогнозирования работы команды? Нет ничего проще. Мы используем принцип “вчерашней погоды”. Считаем сколько в среднем команда выполнила сторипоинов в прошлом и делаем прогноз на будущее. Например, если команда работает 2-х недельными итерациями и за последние 6 итераций выполняла 10, 12, 9, 11, 13, 8 сторипоинтов, то наша средняя производительность, часто называемая велосити (velocity) составляет 10,5 сторипоинтов за итерацию. На эту величину мы и будем опираться при следующем планировании. Обратите внимание, что нам вообще не требуется знать сколько часов команда тратила на ту или иную работу.

Следует учитывать, что команда включает в одну общую оценку всю работу, которую нужно сделать. Например, чтобы выполнить задачу, может понадобиться проработка дизайна, разработка, тестирование, выкладка и сбор метрик на реальных пользователях. Мы не разделяем оценку на части, а даем одну общую, интегральную оценку на всю задачу. В оценивании принимает участие вся команда.

Чем больше оценка, тем больше потенциальных рисков несет в себе задача, тем сложнее ее делать и выше вероятность что задача затянется. Поэтому многие опытные команды используют большую оценку как сигнал, что эта задача слишком большая, брать ее в работу нельзя, и, чтобы снизить риски, нужно ее разбить на несколько задач поменьше. Команды устанавливают себе планку, например не брать в работу задачи больше 13 сторипоинтов и постепенно эту планку снижают.

Оценка в сторипоинтах удобнее тем, что она устойчива к разнице компетенций, включает объем, сложность и риски, а также не привязана к абсолютным значениям (дни, часы), что снижает (хотя и полностью не исключает) возможность давления на команду со стороны менеджмента.

Джеф Сазерленд, один из авторов фреймворка Scrum, в своей статье убедительно доказывает на примере исследований, проводившихся в Microsoft [9], что оценка в сторипоинтах намного точнее абсолютных оценок, значительно сокращает время на планирование, помогает командам точнее предсказывать сроки завершения работы, увеличивает производительность команды и сокращает потери [7]. К таким же выводам приходит Мэри и Том Поппендик, сравнивая производительность команд, использующих разные способы оценки [10].

Другие варианты относительных оценок.

Относительная оценка в сторипоинтах несет в себе потенциальную опасность, что менеджмент продолжит воспринимать эти оценки как обязательства. Менеджеры довольно легко умеют конвертировать относительные оценки в часы (дни) и обратно. Поэтому часто команды используют относительные оценки, которые невозможно перевести в числовые показатели. Например, можно использовать шкалу размеров футболок (S, M, L), корзины (просто, обычно, сложно) или абстрактные единицы (банан, велосипед, мишка Гамми, курица, вертолет).

Шкалой с небольшим количеством элементов удобно пользоваться для быстрой, но менее точной оценки. Практика показывает, что чем больше возможностей выбора, тем больше времени команда будет тратить на споры «это не 2, а 3» и выбор конкретной оценки. А если выбор сводится к «эта история большая, маленькая или средняя?», то и сам процесс оценки проходит гораздо быстрее. При этом точность не слишком падает, так как, как мы помним, оценка – величина всегда неточная, а значит не стоит тратить слишком много времени на попытки получить точное значение. Время вы потратите, но все равно ошибетесь.

#noestimates.

Как мы обсуждали выше, чем выше оценка, тем больше неопределенности несет в себе задача и тем выше риск не успеть ее сделать в запланированные сроки. Поэтому опытные команды со временем вырабатывают навык декомпозировать все задачи на более мелкие. Когда команда работает над задачами, относительный размер которых отличается меньше, чем в 5-10 раз, можно попробовать отказаться от оценок совсем.

Подход #noestimates предлагает вообще отказаться от оценивания задач. Вместо этого мы считаем среднее количество выполненных задач в прошлых итерациях и делаем прогноз в штуках. Например, в среднем за последние 6 итераций команда завершала 8 задач, значит с высокой долей вероятности и в следующей итерации команда сделает около 8 задач. Когда размер задач отличается не больше одного порядка, вы сможете сэкономить огромное количество времени и усилий на планировании, если откажетесь от оценок и начнете просто опираться на количество сделанных задач.

Возможно, вам кажется, что при таком подходе вы сильно потеряете в предсказуемости. Васко Дуарте, автор книги #noestimates, показывает на данных реальных проектов, что предсказательная способность оценивания в штуках отличается от предсказательной способности в сторипоинтах в пределах статистической погрешности [6].

Как правило, команды, научившиеся декомпозировать задачи достаточно мелко, переходят на #noestimates подход быстро и безболезненно.

Заключение

Мы рассмотрели несколько способов оценивания работы команды разработчиков: абсолютные оценки в часах и идеальных часах, относительные оценки в сторипоинтах и других шкалах и подход #noestimates. Не существует идеального способа оценки, подходящего всем командам. Каждый способ обладает своими преимуществами и недостатками. Выбирайте подходящий для вашей команды способ оценивания и пробуйте новые, когда будете готовы.

Литература

1. Кон, М. Agile: оценка и планирование проектов. Москва: Альпина Паблишер. 2018. 460 с.
2. Cohn, M. Succeeding with Agile: Software Development Using Scrum. Addison-Wesley Professional. 2009. 512 с.
3. Duarte, V. NoEstimates: How To Measure Project Progress Without Estimating. OikosofySeries. 2016. 252 с.

4. Iqbal, M. How to Create a Point System for Estimating in Scrum / Сайт: scrum.org, 5 мая 2022. [Электронный ресурс]. URL: https://www.scrum.org/resources/blog/how-create-point-system-estimating-scrum (дата обращения: 03.03.2023).
5. Iqbal, M. When will we get there? How to estimate in Scrum / Сайт: scrum.org, 13 апреля 2022. [Электронный ресурс]. URL: https://www.scrum.org/resources/blog/when-will-we-get-there-how-estimate-scrum (дата обращения: 03.03.2023).
6. Jeffries, R. Story Points Revisited / Сайт: ronjeffries.com, 23 мая 2019. [Электронный ресурс]. URL: https://ronjeffries.com/articles/019-01ff/story-points/Index.html (дата обращения: 03.03.2023).
7. Sutherland, J. Story Points: Why are they better than hours? / Сайт: scruminc.com, 16 мая 2013. [Электронный ресурс]. URL: https://www.scruminc.com/story-points-why-are-they-better-than/?roistat_visit=195302 (дата обращения: 03.03.2023).
8. Dinwiddie, G. Software Estimation without Guessing: Effective Planning in an Imperfect World. Raleigh: The Pragmatic Bookshelf. 2020. 354 с.
9. Williams, L.; Brown, G.; Meltzer, A.; Nagappan, N. Scrum + Engineering Practices: Experiences of Three Microsoft Teams / 2011 International Symposium on Empirical Software Engineering and Measurement, Banff, AB, Канада, 2011, С.: 463-471, DOI: 10.1109/ESEM.2011.65.
10. Poppendieck, M.; Poppendieck, T. Lean Software Development: An Agile Toolkit. Addison-Wesley Professional. 2003. 240 с.

COMPARATIVE ANALYSIS OF DEVELOPMENT TEAM WORK ESTIMATING METHODS

Anton Bevzuk

Engineering Manager of Mindbox LLC

Nizhny Novgorod, Russia

Irina Bevzuk

Senior Lecturer of Foreign Languages Theory and Practice and Linguodidactics Department in Nizhny Novgorod State Pedagogical University of Kozma Minin (Minin University)

Nizhny Novgorod, Russia

Svetlana Tsvetkova

PhD, Associate Professor of Foreign Languages Theory and Practice and Linguodidactics Department in Nizhny Novgorod State Pedagogical University of Kozma Minin (Minin University)
Nizhny Novgorod, Russia

Abstract. It is important for every development team to properly evaluate their work in order to achieve their goals and meet deadlines. There are several methods of performance appraisal, each with its own advantages and disadvantages. Some teams use absolute hour estimates to more accurately estimate the amount of time needed to complete certain tasks. However, this can take a lot of planning time. Relative assessment in storypoints allows you to quickly assess the complexity of tasks and use this indicator as a unit of measurement, but it is not always convenient for large projects and requires a certain skill in evaluating work in relative units of measurement. On the other hand, #noestimates approach means not grading at all, which can be helpful for teams that find grading ineffective or who can't rate their work in hours or storypoints. Ultimately, the choice of method depends on the specifics of the project and the experience of the team.

Keywords: Agile; management; planning; estimating; absolute estimating; relative estimating; story points; noestimates.

JEL codes: M12; O21.

References

1. Cohn, M. (2018) Agile: Project Evaluation and Planning. Moscow: Alpina Publisher. 460 p.
2. Cohn, M. (2009) Succeeding with Agile: Software Development Using Scrum. Addison Wesley Professional. 512 p.
3. Duarte, V. (2016) NoEstimates: How To Measure Project Progress Without Estimating. OikosofySeries. 252 p.
4. Iqbal, M. (2022) How to Create a Point System for Estimating in Scrum. URL: https://www.scrum.org/resources/blog/how-create-point-system-estimating-scrum.
5. Iqbal, M. (2022) When will we get there? How to estimate in Scrum. URL: https://www.scrum.org/resources/blog/when-will-we-get-there-how-estimate-scrum.
6. Jeffries, R. (2019) Story Points Revisited. URL: https://ronjeffries.com/articles/019-01ff/story-points/Index.html.
7. Sutherland, J. (2013) Story Points: Why are they better than hours? URL: https://www.scruminc.com/story-points-why-are-they-better-than/?roistat_visit=195302.

8. Dinwiddie, G. (2020) Software Estimation without Guessing: Effective Planning in an Imperfect World. Raleigh: The Pragmatic Bookshelf. 354 p.
9. Williams, L.; Brown, G.; Meltzer, A.; Nagappan, N. (2011) Scrum + Engineering Practices: Experiences of Three Microsoft Teams / 2011 International Symposium on Empirical Software Engineering and Measurement, Banff, AB, Canada, 2011, P.: 463-471, DOI: 10.1109/ESEM.2011.65.
10. Poppendieck, M.; Poppendieck, T. (2003) Lean Software Development: An Agile Toolkit. Addison Wesley Professional. 240 p.

Contact

Anton Bevzuk
Mindbox LLC
26, Pravdy st., 125124, Moscow, Russia
bevzuk@gmail.com

Irina Bevzuk
Nizhny Novgorod State Pedagogical University
1, st. Ulyanova, 603950, Nizhny Novgorod, Russia
ibevzuk@gmail.com

Svetlana Tsvetkova
Nizhny Novgorod State Pedagogical University
1, st. Ulyanova, 603950, Nizhny Novgorod, Russia
svetlanatsvetkova5@gmail.com